Auddard & Co.

Arithmetic Expression Evaluator in C++

Test Case

Version 1.1

Revision History

Date	Version	Description	Author
11/Nov./23	1.0	About 5-15 test cases were generated on a separate document before the template was posted on Canvas	Audrey Pan, Sriya Annem, Morgan Nguyen, Lauren D'Souza, Ella Nguyen, Tanu Sakaray
29/Nov./23	1.1	The Purpose section was finished and about 15 remaining test cases were generated and tested on the program	Audrey Pan, Sriya Annem, Morgan Nguyen, Lauren D'Souza, Ella Nguyen, Tanu Sakaray

Table of Contents

Test Case

1. Purpose

A test case document for the C++ calculator program serves as a crucial component in ensuring the reliability and functionality of the software. The primary purpose of this document is to outline various scenarios, inputs, and expected outcomes, providing a structured approach to validating the correctness of the implemented code. Test cases are essential as they simulate real-world usage scenarios, helping to identify and fix potential bugs, errors, or unexpected behaviors in the calculator program.

Having a comprehensive set of test cases is important for several reasons. Firstly, it ensures that the calculator functions as intended under different conditions, covering a wide range of inputs and user interactions. This systematic testing process enhances the overall quality of the software by minimizing the likelihood of undetected issues. Furthermore, a well-documented test case suite helps collaboration among team members, allowing for communication of testing requirements and results.

In real-life applications, rigorous testing is imperative to guarantee the reliability and accuracy of software. In the context of this C++ calculator project, creating a test case document is not only a current necessity but an investment in the future. As the codebase evolves and new features are added, having a robust testing framework will streamline the debugging process and ensure that modifications do not make errors.

2. Test case identifier

See table below

3. Test item

See table below

4. Input specifications

See table below

5. Output specifications

See table below

Test Case ID	Test Case Description	Test Data	Expected Results	Actual Results	Pass/Fail
ADD101	Complex Addition with Extraneous Parentheses	(((2 + 3))) + (((1 + 2)))	8	8	Pass
ADD102	Test addition of negative numbers with order of operations	17 + ((-3)) + (+9)	23	23	Pass
ADD103	Test addition of integer with float	3 + 4.5	7.5	7.5	Pass
COMPLEX101	Mixture of exponents, parentheses and mixed	3 ^ 2 * (5 - 2) + 4 / 2	29	29	Pass

	operations				
COMPLEX102	Multiple levels of parentheses and mixed operations	(2 + 3) * (4 / 2)	10	10	Pass
COMPLEX103	Exponents, division, and subtraction	(2 ^ 3) * 4 / (5 - 2)	10.6667	10.6667	Pass
DIV101	Division of Negative Numbers	(-10)/ (-2)	5	5	Pass
EDGE101	Division by zero	0 / 5	0	0	Pass
EXP101	Complex Exponentiation and Parentheses	(3 ^ (4 - 2)) * (8 / 2)	36	36	Pass
EXP102	Exponentiation and Large Numbers	2 ^ 10	1024	1024	Pass
EXP103	Exponent with fraction (squareroot)	9 ^ (1 / 2) + ((2 ^ 3) - 4) * (10 / 2)	23	23	Pass
FLOAT107	Division with floats	(1.5 * 3) / (0.5 + 0.5)	4.5	4.5	Pass
FLOATP101	Floating Point Inputs	5.5 + 2.2 * 3.1	12.32	12.32	Pass
FLOATP102	Floating Point with Exponents	2.0 ^ (3.5)	11.3137	11.3137	Pass
FLOATP103	Floating Point with Division	10.8 / 3.6	3	3	Pass
FLOATP104	Floating Points and Nested Parentheses	(3.7 + (5.2 - 1.1)) * 2.5	19.5	19.5	Pass
FLOATP105	Negative Floating Points	-1.5 * 2.2	-3.3	-3.3	Pass
FLOATP106	Floating Point Exponentiation with Parentheses	(5.1 - 2.6) ^ 2	6.25	6.25	Pass
INVALID101	Multiple invalid characters	1abc + abc	Error	Error	Pass
INVALID102	Test not having enough operands	5 + + 5	Error	Error	Pass
INVALID103	Test handling of trailing operators	2 + 3 *	Error	Error	Pass
MOD101	Modulo of Positive	33 % 4	1	1	Pass

	Numbers				
MULT101	Test multiplication of negative numbers and order of operations	4 + (-3) * 2	-2	-2	Pass
PAREN101	Combination of Extraneous and Necessary Parentheses	(((((5 - 3))) * (((2 + 1))) + ((2 * 3))))	12	12	Pass
PAREN102	Nested Parentheses with Exponents	(((2 ^ (1 + 1)) + ((3 - 1) ^ 2)) / ((4 / 2) % 3))	4	4	Pass
PAREN103	Extraneous Parentheses with Division	((9 + 6)) / ((3 * 1) / (((2 + 2))) - 1)	-60	-60	Pass
PAREN104	Combining Unary Operators with Parentheses	-(2) * (3) - (-4) / (-5)	-6.8	-6.8	Pass
SUB101	Test handling of subtraction and negative numbers	4 - 2 - (-9)	11	11	Pass
SUB102	Test subtraction with positive numbers and order of operations	-12 - (+2) * 13 - (+7)	-45	-45	Pass